Deep Generative Models

Shenlong Wang
Overview

- Why unsupervised learning?
- Old-school unsupervised learning
  - PCA, Auto-encoder, KDE, GMM
- Deep generative models
  - VAEs, GANs
Unsupervised Learning

- No labels are provided during training
- General objective: inferring a function to describe hidden structure from unlabeled data
  - Density estimation (continuous probability)
  - Clustering (discrete labels)
  - Feature learning / representation learning (continuous vectors)
  - Dimension reduction (lower-dimensional representation)
  - etc.
Why Unsupervised Learning?

- Density estimation: estimate the probability density function $p(x)$ of a random variable $x$, given a bunch of observations $\{X_1, X_2, \ldots\}$.

Credit: BallR

2D density estimation of Stephen Curry’s shooting position
Why Unsupervised Learning?

- Density estimation: estimate the probability density function $p(x)$ of a random variable $x$, given a bunch of observations \{X_1, X_2, \ldots\}
Why Unsupervised Learning?

- Clustering: grouping a set of input \{X_1, X_2, \ldots\} in such a way that objects in the same group (called a cluster) are more similar.
Why Unsupervised Learning?

- Feature learning: a transformation of raw data input to a representation that can be effectively exploited in machine learning tasks.

2D topological visualization given the input how similar players are with regard to points, rebounds, assists, steals, rebounds, blocks, turnovers and fouls.

Credit: Ayasdi
Why Unsupervised Learning?

- Dimension reduction: reducing the number of random variables under consideration, via obtaining a set of principal variables

Principle component analysis over players trajectory data

Credit: Bruce, Arxiv 2016
Principle Component Analysis (PCA)

An algorithm that conducts dimension reduction

Intuition:

- Finds the lower-dimension projection that minimizes reconstruction error
- Keep the most information (maximize variance)

See more details in Raquel's CSC411 slides:
http://www.cs.toronto.edu/~urtasun/courses/CSC411_Fall16/14_pca.pdf
Principle Component Analysis (PCA)

An algorithm that conducts dimension reduction

Intuition:
- Finds the lower-dimension projection that minimizes reconstruction error
- Keep the most information (maximize variance)

Algorithm:
- Conduct eigen decomposition
- Find K-largest eigenvectors
- Linear projection with the matrix composed of K eigenvectors

See more details in Raquel's CSC411 slides:
http://www.cs.toronto.edu/~urtasun/courses/CSC411_Fall16/14_pca.pdf
Auto-encoder

A neural network that the output is the input itself.

Intuition:

- A good representation should keep the information well (reconstruction error)
- Deep + nonlinearity might help enhance the representation power

\[
\min_{w_1, w_2} \| x_i - g(f(x_i; w_1); w_2) \|^2_2
\]
Auto-encoder

A neural network that the output is the input itself.

Intuition:

- A good representation should keep the information well (reconstruction error)
- Deep + nonlinearity might help enhance the representation power

\[
\min_{w_1, w_2} \| x_i - g(f(x_i; w_1); w_2) \|^2_2
\]

Credit: LeCun
Auto-encoder

A neural network that the output is the input itself.

10-dimensional Auto-encoder feature embedding based on players shooting tendency

Credit: Wang et al. 2016 Sloan Sports Conference
Kernel Density Estimation (KDE)

A nonparametric way to estimate the probability density function of a random variable

Intuition:

- Point with more neighbouring samples have higher density
- Smoothed histogram, centered at data point

\[
f(x) = \frac{1}{N} \sum_i \frac{1}{h_i} K\left(\frac{x - x_i}{h_i}\right)
\]

Kernel function, measures the similarity
Kernel Density Estimation (KDE)

A nonparametric way to estimate the probability density function of a random variable

Applications:

- Visualization
- Sampling

$$f(x) = \frac{1}{N} \sum_i^N \frac{1}{h_i} K \left( \frac{x - x_i}{h_i} \right)$$

Shooting heat map of Lamarcus Aldridge 2015-2016. Credit: Squared Statistics
Generative models

Task: generate new samples follows the same probabilistic distribution of a given training dataset.
Generative models

Task: generate new samples follows the same probabilistic distribution of a given a training dataset

\[ Z \sim \mathcal{N}(0, 1) \]
Generative models

Task: generate new samples follows the same probabilistic distribution of a given training dataset.

Note: sometimes it’s fine if we cannot estimate the explicit form of $p(x)$, since it might be over complicated.
Variational Auto-encoder (VAE)

Intuition: given a bunch of random variables that can be sampled easily, we can generate random samples following other distributions, through a complicated non-linear mapping $x = f(z)$

$$f(z) = z/10 + z/\|z\|$$

Image Credit: Doersch 2016
Variational Auto-encoder (VAE)

Intuition: given a bunch of random variables that can be sampled easily, we can generate some new random samples through a complicated non-linear mapping $x = f(z)$

$Z \sim \mathcal{N}(0, 1)$

$X \sim \mathcal{N}(f(z; \theta), \sigma^2 I)$

Image Credit: Doersch 2016
Variational Auto-encoder (VAE)

Intuition: given a bunch of random variables, we can generate some new random samples through a complicated non-linear mapping $x = f(z)$.
Variational Auto-encoder (VAE)

You can consider it as a decoder!
Variational Auto-encoder (VAE)

How do we learn the parameters of decoder network?
Variational Auto-encoder (VAE)

Review: Marginalization

\[ p(x) = \int p_{\theta}(x|z)p_{\theta}(z)dz \]
Variational Auto-encoder (VAE)

Review: Marginalization

\[ p(x) = \int p_{\theta}(x|z)p_{\theta}(z) \, dz \]

\([0, 0.2, -0.5] \]

Image Credit: Doersch 2016
Variational Auto-encoder (VAE)

Learning objective: maximize the log-probability

$$\max_\theta \sum_i \log p_\theta(x_i)$$

$p_\theta(x) = \int p_\theta(x|z)p_\theta(z)dz$

Training images should have high probability
Variational Auto-encoder (VAE)

Learning objective: maximize the log-probability

$$\max_{\theta} \sum_i \log p_{\theta}(x_i)$$

$$p_{\theta}(x) = \int p_{\theta}(x|z)p_{\theta}(z)dz$$

Integration over a neural network. Difficult!
Variational Auto-encoder (VAE)

Learning objective: maximize the log-probability

\[ \max_{\theta} \sum_{i} \log p_{\theta}(x_i) \]

\[ p_{\theta}(x) = \int p_{\theta}(x|z)p_{\theta}(z)dz \]

Integration over a neural network. Difficult!

Quiz: Why not do this?

\[ \log p_{\theta}(x) \approx \log \frac{1}{N} \sum_{j} p_{\theta}(x|z_j) \]

Image Credit: Doersch 2016
Variational Auto-encoder (VAE)

Learning objective: maximize the log-probability

$$\max_{\theta} \sum_i \log p_\theta(x_i)$$

$$p_\theta(x) = \int p_\theta(x|z)p_\theta(z)dz$$

many sampled $z$ will have a close-to-zero $p(x|z)$

Quiz: Why not do this?

$$\log p_\theta(x) \approx \log \frac{1}{N} \sum_j p_\theta(x|z_j)$$

Image Credit: Doersch 2016
Variational Auto-encoder (VAE)

Learning objective: maximize variational lower-bound

$$\log p_\theta(x_i) \geq \mathbb{E}_{q(z)}[\log p_\theta(x_i | z)] - KL[q(z) || p_\theta(z)]$$

Quiz: How to choose a good proposal distribution?
Variational Auto-encoder (VAE)

Learning objective: maximize variational lower-bound

$$\log p_\theta(x_i) \geq \mathbb{E}_{q(z)}[\log p_\theta(x_i|z)] - KL[q(z)||p_\theta(z)]$$

Proposal distribution

Variational lower-bound

Quiz: How to choose a good proposal distribution?

- Easy to sample
- Differentiable wrt parameters
- Given a training sample $X$, the sampled $z$ is likely to have a non-zero $p(x|z)$
Variational Auto-encoder (VAE)

Learning objective: maximize variational lower-bound

\[
\log p_\theta(x_i) \geq \mathbb{E}_{q_\phi(z|x_i)}[\log p_\theta(x_i|z)] - KL[q_\phi(z|x_i) || p_\theta(z)]
\]

Answer: Another neural network + Gaussian to approximate the posterior!
Variational Auto-encoder (VAE)

Learning objective: maximize variational lower-bound

\[
\log p_\theta(x_i) \geq \mathbb{E}_{q_\phi(z|x_i)}[\log p_\theta(x_i|z)] - KL[q_\phi(z|x_i)||p_\theta(z)]
\]

Reconstruction error:
- Training samples have higher probability

Prior:
- Proposal distribution should be like Gaussian
Variational Auto-encoder (VAE)

Learning objective: maximize variational lower-bound

\[
\log p_\theta(x_i) \geq \mathbb{E}_{q_\phi(z|x_i)}[\log p_\theta(x_i|z)] - KL[q_\phi(z|x_i) \parallel p_\theta(z)]
\]

- KL-Divergence: closed-form and differentiable if both are Gaussians
- Reconstruction error: approximate by just sampling one \(z\)

Credit: Doersch
Variational Auto-encoder (VAE)

Why it is the variational lower-bound?

\[
\log p_\theta(x) = \log \int p_\theta(x|z)p_\theta(z)dz
\]

\[
\log p_\theta(x) = \log \int p_\theta(x|z)\frac{p_\theta(z)}{q(z)}q(z)dz
\]

\[
\log p_\theta(x) \geq \int q(z) \log \left( \frac{p_\theta(x|z)}{q(z)} \right) dz
\]

\[
\log p_\theta(x) \geq \int q(z) \log p_\theta(x|z)dz - \int q(z) \log \frac{p_\theta(z)}{q(z)}dz
\]

\[
\log p_\theta(x) \geq \mathbb{E}_{q(z)}[\log p_\theta(x|z)] - KL[q(z)\|p_\theta(z)]
\]

Jenson inequality

\[
\log \int p(x)g(x)dx \geq \int p(x) \log g(x)dx
\]

Kingma et al. 2014
Variational Auto-encoder (VAE)

The whole learning structure

Input Image $x$ → Encoder network → Code $z$ → Decoder network → Reconstruction

KL-Divergence
Reconstruction Loss
Variational Auto-encoder (VAE)

Results

(a) Learned Frey Face manifold

(b) Learned MNIST manifold

Kingma et al. 2014
Variational Auto-encoder (VAE)

VAE Demo

Kingma et al. 2014
Generative Adversarial Network (GAN)

Code $z$ → Generator → Generated Image
Generative Adversarial Network (GAN)

Code $z$ → Generator → Generated Image → Discriminator

Fake → Real

Training Image
Generative Adversarial Network (GAN)

Intuitions

Crook
Generative Adversarial Network (GAN)

Intuitions

Generator

Teller
Generative Adversarial Network (GAN)

Intuitions

Generator

Teller

- Mini characters: tiny "RMB" and "100" printed repeatedly
- The "security line" can be seen if viewed at 180 degrees against the light
- An invisible "100", can only be seen when viewed at 180 degrees against the light.
- Watermark: transparent Mao that appears when viewed against the light
- Color-Changing ink: when adjusting the viewing angle, the color changes from green to blue
- Immisible "100"
- Yin and Yang: the two patterns form a complex image of an ancient coin
- Mao's visage, the braille, and name of the bank have an uneven feel when rubbed
Generative Adversarial Network (GAN)

Intuitions

Crook

Teller
Generative Adversarial Network (GAN)

Intuitions:

- Generator tries the best to cheat the discriminator by generating more realistic images.
- Discriminator tries the best to distinguish whether the image is generated by computers or not.
Generative Adversarial Network (GAN)

Objective function:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [1 - \log D(G(z))]$$

For each iteration:

- Sample a mini-batch of fake images and true images
- Update G using back-prop
- Update D using back-prop

Very difficult to optimize:

- Min-max problem: finding a saddle point instead of a local optimum, unstable
Generative Adversarial Network (GAN)

- Code $z$
- Generator
- Generated Image
- Discriminator
- Cross-Entropy
- Training Image
GANs for face and bedroom

Credit: Denton
GANs for Japanese Anime

Credit: Radford
GANs for Videos

Credit: Vondrick
GANs for Image Upsampling

- **bicubic**
  - 21.59dB/0.6423

- **SRResNet**
  - 23.53dB/0.7832

- **SRGAN**
  - 21.15dB/0.6868

Credit: Ledig
Generative Adversarial Network (GAN)

Extensions:

- DCGANs: some hacks that work well
- LAPGANs: coarse-to-fine conditional generation through Laplacian pyramids
- f-GANs: more general GANs with different loss other than cross-entropy
- infoGANs: additional objective that maximize mutual-information between the latent and the sample
- EBGANs: Discriminative as energy functions
- GVMs: using GANs as an energy term for interactive image manipulation
- Conditional GANs: not random z, instead z is some data from other domain
- ...

Generative Adversarial Network (GAN)

Hacks:

- How to train a GAN?
- 17 hacks that make the training work.
- https://github.com/soumith/gan_hacks
Generative Adversarial Network (GAN)

GAN Demo
GANs vs VAEs

GANs:

- High-quality visually appealing result
- Difficult to train
- The idea of adversarial training can be applied in many other domains

VAEs:

- Easy to train
- Blurry result due to minimizing the MSE based reconstruction error
- Nice probabilistic formulation, easy to introduce prior
Demos

VAEs:


GANs:

- [https://github.com/ericjang/genadv_tutorial/blob/master/genadv1.ipynb](https://github.com/ericjang/genadv_tutorial/blob/master/genadv1.ipynb)
- [https://gist.github.com/wiseodd/b2697c620e39cb5b134bc6173cfe0f56](https://gist.github.com/wiseodd/b2697c620e39cb5b134bc6173cfe0f56)
References

[12] Wikipedia "Principal component analysis"
Thanks